

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-055092

(43)Date of publication of application : 27.02.1996

(51)Int.Cl.

G06F 15/16

G06F 9/38

G06F 9/46

(21)Application number : 06-188147

(71)Applicant : NEC CORP

(22)Date of filing : 10.08.1994

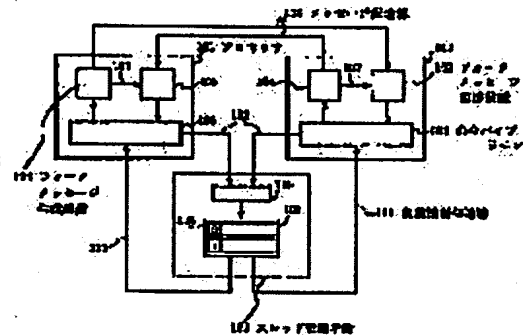
(72)Inventor : MOTOMURA MASATO
TORII ATSUSHI

(54) PROCESSOR SYSTEM AND ITS CONTROL METHOD

(57)Abstract:

PURPOSE: To provide a dynamic parallelism control mechanism and its control method which extract the excessive parallelism to prevent the degradation of performance with respect to the processor system which makes the thread fork to another processor by a fork instruction to perform the parallel processing.

CONSTITUTION: A thread management means 121 is provided besides plural processors 101 to realize the processor system. The thread management means 121 is provided with a load state table 109, and this table 109 consists of plural load state counters 108 which show the load states of processors 101. Load state counters 108 are accessed to find the dynamic load states of processors 101, and they are used to control the parallelism.



LEGAL STATUS

[Date of request for examination] 10.08.1994

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 2684993

[Date of registration] 15.08.1997

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

(19)日本国特許庁 (J P)

(12) 特 許 公 報 (B 2)

(11)特許番号

第2684993号

(45)発行日 平成9年(1997)12月3日

(24)登録日 平成9年(1997)8月15日

(51)Int.Cl. ⁸	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16	3 8 0		G 0 6 F 15/16	3 8 0 Z
9/38	3 1 0		9/38	3 1 0 J
9/46	3 4 0		9/46	3 4 0 D

請求項の数9 (全 9 頁)

(21)出願番号	特願平6-188147	(73)特許権者	000004237 日本電気株式会社 東京都港区芝五丁目7番1号
(22)出願日	平成6年(1994)8月10日	(72)発明者	木村 真人 東京都港区芝五丁目7番1号 日本電気株式会社内
(65)公開番号	特開平8-55092	(72)発明者	島居 淳 東京都港区芝五丁目7番1号 日本電気株式会社内
(43)公開日	平成8年(1996)2月27日	(74)代理人	弁理士 京本 直樹 (外2名)
		審査官	石井 茂和
		(56)参考文献	特開 平3-40034 (J P, A) 特開 平5-313922 (J P, A) 特開 昭56-66633 (J P, A)

(54)【発明の名称】 プロセッサシステムとその制御方法

1

(57)【特許請求の範囲】

【請求項1】複数の命令から構成されるスレッドを外部へフォークするスレッドフォーク手段と、外部からフォークされた前記スレッドを発行するスレッド発行手段と、発行された前記スレッド内の複数の前記命令を順次処理する命令パイプラインとを備え、スレッドのフォークを指示するフォーク命令を制御方法として備えたプロセッサを複数個用い、ある前記プロセッサにおいて前記フォーク命令を実行することにより前記スレッドフォーク手段を介して前記スレッドをフォークし、別の前記プロセッサにおいてフォークされた前記スレッドを前記スレッド発行手段を介して発行して前記スレッドの複数の前記命令を前記命令パイプラインで実行することにより並列処理を実現するプロセッサシステムであって、これら複数個の前記プロセッサ間でスレッド管理手段を共有

2

し、前記スレッド管理手段は前記プロセッサシステムの負荷状態を示す負荷状態カウンタを保有し、しかもプロセッサの命令セットの一部としてスレッドの数を制限する命令を組み込み、この命令を用いて前記スレッド管理手段を駆動することを特徴とするプロセッサシステム。
【請求項2】プロセッサの命令セットの一部としてスレッドの数を制限する命令として、少なくともカウンタロック命令、カウンタアンロック命令を有する請求項1記載のプロセッサシステム。

10

【請求項3】請求項1に記載のプロセッサシステムの制御方法であって、負荷状態カウンタロック命令を規定し、ある前記プロセッサが前記負荷状態カウンタロック命令を実行することにより、当該の前記プロセッサが前記スレッド管理手段内の前記負荷状態カウンタをロックし、他の前記プロセッサが前記負荷状態カウンタの値を

3

変更できないようにすることを特徴とするプロセッサシステムの制御方法。

【請求項4】請求項1に記載のプロセッサシステムの制御方法であって、負荷状態カウンタアンロック命令を規定し、ある前記プロセッサが前記負荷状態カウンタアンロック命令を実行することにより、当該の前記プロセッサが前記スレッド管理手段内の前記負荷状態カウンタをアンロックし、他の前記プロセッサが前記負荷状態カウンタの値を変更できるようにすることを特徴とするプロセッサシステムの制御方法。

【請求項5】請求項1に記載のプロセッサシステムの制御方法であって、前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に、前記負荷状態カウンタをアンロックすることを特徴とするプロセッサシステムの制御方法。

【請求項6】請求項1、3、4または5に記載のプロセッサシステムの制御方法であって、前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に前記負荷状態カウンタの値を増やすことと、ある前記プロセッサが前記スレッドの実行を終了する際に前記負荷状態カウンタを減じることを特徴とするプロセッサシステムの制御方法。

【請求項7】請求項1、3、4、5または6に記載のプロセッサシステムの制御方法であって、前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に、前記スレッド管理手段を排他的に参照し、前記負荷状態カウンタの値が一定の負荷状態以下を示しているかどうかを条件として前記スレッドのフォークが可能かどうかを判定し、フォークが可能な場合は前記スレッドのフォークを行ない、フォークが不可能な場合は当該する前記プロセッサ上で前記スレッドを逐次的に実行することを特徴とするプロセッサシステムの制御方法。

【請求項8】請求項1、3、4、5、6または7に記載のプロセッサシステムの制御方法であって、前記フォーク命令を実行することにより前記プロセッサがスレッドをフォークする際に、フォークされた前記スレッドが前記プロセッサシステムを構成する複数の前記プロセッサのうちどの前記プロセッサで実行されるかを、前記負荷状態カウンタの値を参照することにより、前記スレッド管理手段が決定することを特徴とするプロセッサシステムの制御方法。

【請求項9】請求項1、3、4、5、6、7または8に記載のプロセッサシステムの制御方法であって、スレッド確保命令を規定し、前記スレッド確保命令を実行することにより、前記プロセッサが前記スレッド管理手段を参照して前記スレッドをフォークすることが可能かどうかを確認し、可能である場合は前記スレッド管理手段内

4

の前記負荷状態カウンタの値を増やすことを特徴とするプロセッサシステムの制御方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は情報処理システムの中核をなすプロセッサに関し、更に複数のプロセッサ間でスレッドをフォークしながらプログラムの並列処理を実行するプロセッサシステムに関する。

【0002】

10 【従来の技術】一般に、プロセッサは、命令列で構成されるプログラムから一つの命令を読み込み、その命令を解釈して指示された処理を実行する、という手順を繰り返すことにより、プログラムが意図するところの全体処理を実現する。この際、プログラムの実行に必要なデータは適宜、命令の指示に従いながらメモリ装置から読み込まれ、あるいは実行結果がメモリ装置に書き込まれる。従来技術としてのプロセッサの基本的な動作は、このようにプログラム内に記述された命令の指示に従って内部的な処理を順次行なっていくものであり、このよう

20 な処理を担当する部分を命令パイプラインと総称する。【0003】一方、このようなプロセッサを使用してプロセッサシステムを構成するにあたっては、上記のような命令パイプラインによる内部的な動作の他に、プロセッサの外部に接続された機器の制御を行なうため、あるいは外部の機器からの制御を受け入れるために外部とプロセッサ間のインタフェースを設ける必要がある。プロセッサから外部機器の制御を行なう従来技術としては、メモリマップドI/O方式や、特殊命令による制御方式が知られている。一方、外部からプロセッサの制御を行なう従来技術としては、割り込み処理技術が知られている。これらの従来技術についてはコンピュータ科学の教科書、例えば“Computer Organization & Design: The Hardware Software Interface”, David Patterson and John Hennessy, Morgan Kaufmann Publishers Inc. p. 566などに詳しく記載されている。

40 【0004】一般に、プロセッサに対して外部より割り込みが通知されると、ユーザープログラムから割り込み処理用のプログラム(割り込みハンドラと呼ばれる)へと当該プロセッサ上の実行プログラムが切り替わり、割り込みハンドラが通知された割り込みに対応した処理を行ない、元のユーザープログラムに戻るかまたは必要に応じて更に別のプログラムを起動する、といった手順が実行される。このように実行するプログラムが切り替わるため、これにともなってプログラムの実行環境自体も切り替わる。ここでプログラムの実行環境とはレジスタファイルに記憶されているデータやプログラムカウンタ、あるいはプロセスコントロールブロックの中身など

50

であり、切り 替わり 時には、後ほど再度使用する
に、これらの実行環境をメモリ 装置内に退避する必要が
ある。このようなオーバーヘッドが存在するため、割り
込み処理は一般には長大な時間を有することが知られて
いる。このような問題については、“The Inte
raction of Architecture and Operating System Desig
n”, Proceedings of Fourth
International Conference
on Architectural Support f
or Programming Languages
and Operating Systems, pp.
108-120, Thomas E. Anderson,
Henry M. Levy, Brian N. Bershad and Edward D. Lazowsk
a などに詳しく 記載されている。

【0005】一方、複数のプロセッサを用いて並列処理
を実現するプロセッサシステムにおいては、プロセッサ
間の通信が頻繁に行なわれる。これは処理の分担を図る
ための通信や同期をとるための通信が多発するためである
。従来のプロセッサを使用したプロセッサシステムでは
上記のプロセッサ間通信を割り 込み処理技術を用いて
実現していた。このため、プロセッサ間通信処理に大き
なオーバーヘッドが存在し、これが並列処理により性能
を向上する上でのボトルネックとなっていた。
【0006】このような問題を解決するため、従来技術
の一つとしてマルチスレッドアーキテクチャと呼ばれる
プロセッサのアーキテクチャが提案されている。ここで
スレッドとは複数の命令から構成される命令列であり、
一つのプログラムは複数のスレッドの集合として定義さ
れるものとする。マルチスレッドアーキテクチャにおい
ては、スレッド単位で処理を複数のプロセッサに分割
し、これらのスレッドを並列に処理する。このために、
マルチスレッドアーキテクチャは、一般に、あるプロセ
ッサ上で実行されているスレッドが別のプロセッサ上に
新たにスレッドを生成するための機構及び命令、更には
あるプロセッサ上で実行されているスレッドと別のプロ
セッサ上で実行されているスレッドとの間で同期をとる
ための機構及び命令などを特徴として備えている。本発
明はスレッド生成に関する新規技術を提案するものであ
るので、以下ではマルチスレッドアーキテクチャの技術
について説明する。

【0007】図8に従来技術のマルチスレッドアーキテ
クチャに基づくプロセッサシステムの構成例を示す。図
8においてプロセッサシステムは複数のプロセッサ81
から構成されており、それぞれのプロセッサ81は命令
パイプライン85とフォークメッセージ生成装置84、
フォークメッセージ記憶装置83から構成される。

【0008】ここで他のプロセッサ81上に新たなスレ
ッドを生成することを「スレッドをフォークする」と呼

ぶことにする。各プロセッサ81のアーキテクチャはフ
ォーク命令をその命令セットの中に備えている。ここで
フォーク命令とは、スレッドのフォークを指示する命令
であり、この命令をあるプロセッサ81の命令パイプ
ライン85で実行することによりスレッドがフォークされ
る。一般に、フォーク命令は、(1)どのプロセッサ8
1にスレッドをフォークするか、(2)どのようなスレ
ッドをフォークするか、また(3)どのようなデータに
対する処理をフォークするか、などに関する指示を引数
として与えられる命令である。一般に(1)の指示はプ
ロセッサの番号をダイレクトに与えるか、あるいは別の
引数の値を解釈するなどの手段により与えられる。

(2)の指示はスレッドの先頭命令アドレス、(3)の
指示は当該スレッドが使用するスタック領域の先頭アド
レスとして与えられることが多い。

【0009】あるプロセッサ81上でフォーク命令が実
行されると、フォークメッセージがフォーク先として指
定されたプロセッサ81にメッセージ伝達線86を介し
て送られ、当該プロセッサのフォークメッセージ記憶装
置83に受信され、格納される。フォークメッセージは
フォーク命令の引数として与えられた情報を含むメッ
セージである。フォークメッセージを受信したプロセッサ
81は、フォークメッセージ記憶装置83でフォークメ
ッセージを解釈し、命令パイプライン85を用いて指定
されたスレッドの実行を開始する。一般に、フォークメ
ッセージを受信したプロセッサ81が他のスレッドを実
行中の場合が考えられる。このような場合に対応するた
め、フォークメッセージ記憶装置83は複数のフォーク
メッセージを記憶できる機構を持つ必要がある。すなわ
ち、プロセッサ81上であるスレッドの実行が終了する
と、スレッドメッセージ記憶装置83に記憶された複数
のフォークメッセージの中から一つをあるスケジュー
ル規則に従って選び、選ばれたフォークメッセージによ
り指定されたスレッドの実行を命令パイプライン85を用
いて開始する。ここでスケジュール規則とは、単純なフ
ァーストインファーストアウトやラストインファースト
アウトに基づくものなどや、プライオリティレベルなど
によるものなどがある。

【0010】以上簡単に説明したマルチスレッドアーキ
テクチャの代表的な例としては、“T. A. Multi
-threaded Massively Parallel Architecture”, Procee
dings of 19th International Symposium on Computer Ar
chitecture, R. S. Nikhil, G.
M. Papadopoulos and Arvin
d, pp. 156-167で発表されているアーキテ
クチャが挙げられる。なお、本アーキテクチャにおいて
は、上記フォーク命令をスタート命令と呼んでいる。

【0011】以上説明したようなマルチスレッドアーキ

テクチャは、複数のプロセッサ間で並列処理を行なう際に、プロセッサ間通信のオーバーヘッドを大幅に削減することが出来る。このためより粒度の細かい単位で処理を分割し、プログラムに存在する細かい粒度の並列性を有効に利用することが出来る、という利点がある。

【0012】

【発明が解決しようとする課題】以上従来の割り込み処理技術と比べた場合のマルチスレッドアーキテクチャ技術の利点を説明したが、この技術にも大きな欠点がある。それは並列性の制御が困難であるという点である。

【0013】図8から明らかなように、従来技術のマルチスレッドアーキテクチャではあるプロセッサ81がスレッドをフォークすると、受信側のプロセッサ81は対応するフォークメッセージを受信し、格納する以外の手順をとることが出来ない。すなわち送信側のプロセッサ81は一方的にフォークメッセージを送ることができ、一方受信側のプロセッサ81は否応なくこれを受信しなければならない。例えば繰り返し数が巨大なループの各イテレーションをスレッドとしてフォークするような場合などから明らかなように、このようなアーキテクチャ上の制約は不必要な並列性を抽出してしまうという状況を招き、以下のような点で性能が低下してしまうことになる。

【0014】・フォークメッセージ記憶装置81はある一定数のフォークメッセージしか記憶できない。上記のような原因によりこの数を越えるフォークメッセージを受信することを余儀なくされた場合、割り込み処理などにより、フォークメッセージ記憶装置81内に記憶されたフォークメッセージをメインメモリに退避する必要がある。これは大きなオーバーヘッドとなる。

【0015】・大量のスレッドがフォークされた場合、多くのスレッドはかなり後になるまで実行されることがない。よって、これらのスレッドと同期をとる必要があるスレッドは長い時間同期待ちをしてしまうことになる。これによりプロセッサの有効利用率が低下し、処理性能が低下してしまう。

【0016】なお上では説明を割愛したが、従来技術のままで、フォークをしても性能の向上が見込めない場合、割り込み処理によりフォークをキャンセルする方法により送信側が一方的にフォークするという状況を改めることが出来る。しかし、このような方法では割り込み処理によるオーバーヘッドが大きいので、いずれにしても性能の低下は避けることが出来ない。

【0017】以上説明したように、従来技術のマルチスレッドアーキテクチャに基づくプロセッサを用いたプロセッサシステムにおいては、並列性を制御することが困難なため、プロセッサシステムの負荷状況によっては、それぞれのプロセッサを有効利用するために必要とされる以上にスレッドをフォークしてしまうことにより、却って性能低下を招くという問題が生じる。

【0018】本発明の目的は、動的な並列性制御手段とその制御方法を提案することによってこれらの問題を解決し、スレッドを利用した並列処理の実行を効率化することにある。

【0019】

【課題を解決するための手段】本発明によるプロセッサ及びプロセッサシステムは、複数の命令から構成されるスレッドを外部へフォークするスレッドフォーク手段と、外部からフォークされた前記スレッドを発行するスレッド発行手段と、発行された前記スレッド内の複数の前記命令を順次処理する命令パイプラインとを備え、スレッドのフォークを指示するフォーク命令を制御方法として備えたプロセッサを複数個用い、ある前記プロセッサにおいて前記フォーク命令を実行することにより前記スレッドフォーク手段を介して前記スレッドをフォークし、別の前記プロセッサにおいてフォークされた前記スレッドを前記スレッド発行手段を介して発行して前記スレッドの複数の前記命令を前記命令パイプラインで実行することにより並列処理を実現するプロセッサシステムであって、これら複数の前記プロセッサ間でスレッド管理手段を共有し、前記スレッド管理手段は前記プロセッサシステムの負荷状態を示す負荷状態カウンタを保有することを特徴とするプロセッサシステムとして構成される。

【0020】本発明のプロセッサ及びプロセッサシステムにおいて、本発明の第1のプロセッサの制御方法は負荷状態カウンタロック命令を規定し、ある前記プロセッサが前記負荷状態カウンタロック命令を実行することにより、当該の前記プロセッサが前記スレッド管理手段内の前記負荷状態カウンタをロックし、他の前記プロセッサが前記負荷状態カウンタの値を変更できないようにすることを特徴とする前記プロセッサの制御方法を用い、また負荷状態カウンタアンロック命令を規定し、ある前記プロセッサが前記負荷状態カウンタアンロック命令を実行することにより、当該の前記プロセッサが前記スレッド管理手段内の前記負荷状態カウンタをアンロックし、他の前記プロセッサが前記負荷状態カウンタの値を変更できるようにすることを特徴とする前記プロセッサの制御方法を用い、前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に、前記負荷状態カウンタをアンロックすることを特徴とする前記プロセッサの制御方法を用いる。

【0021】本発明のプロセッサ及びプロセッサシステムにおいて、本発明の第2のプロセッサの制御方法は前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に前記負荷状態カウンタの値を増やすことと、ある前記プロセッサが前記スレッドの実行を終了す

る際に前記負荷状態カウンタを減じることを特徴とする前記プロセッサシステムの制御方法を用いる。

【0022】本発明のプロセッサ及びプロセッサシステムにおいて、本発明の第3のプロセッサの制御方法は前記フォーク命令を実行することにより前記プロセッサが前記スレッドフォーク手段を介して前記スレッドをフォークする際に、前記スレッド管理手段を排他的に参照し、前記負荷状態カウンタの値が一定の負荷状態以下を示しているかどうかを条件として前記スレッドのフォークが可能かどうかを判定し、フォークが可能な場合は前記スレッドのフォークを行ない、フォークが不可能な場合は当該する前記プロセッサ上で前記スレッドを逐次的に実行することを特徴とする前記プロセッサの制御方法を用いる。

【0023】本発明のプロセッサ及びプロセッサシステムにおいて、本発明の第4のプロセッサの制御方法は前記フォーク命令を実行することにより前記プロセッサがスレッドをフォークする際に、フォークされた前記スレッドが前記プロセッサシステムを構成する複数の前記プロセッサのうちの前記プロセッサで実行されるかを、前記負荷状態カウンタの値を参照することにより、前記スレッド管理手段が決定することを特徴とする前記プロセッサの制御方法を用いる。

【0024】本発明のプロセッサ及びプロセッサシステムにおいて、本発明の第5のプロセッサの制御方法はスレッド確保命令を規定し、前記スレッド確保命令を実行することにより、前記プロセッサが前記スレッド管理手段を参照して前記スレッドをフォークすることが可能かどうかを確認し、可能である場合は前記スレッド管理手段内の前記負荷状態カウンタの値を増やすことを特徴とする前記プロセッサの制御方法を用いる。

【0025】

【実施例】図1は本発明によるプロセッサシステムの第1の実施例を示すブロック構成図である。図1において、プロセッサシステムは複数のプロセッサ101から構成されており、プロセッサ101は命令パイプライン105とフォークメッセージ生成装置104、フォークメッセージ記憶装置103から構成される。フォークメッセージ生成装置104とフォークメッセージ記憶装置103は、メッセージ伝達線106によって接続されている。自プロセッサ内のフォークメッセージ生成装置104とフォークメッセージ記憶装置103はメッセージ伝達線107により接続されている。

【0026】はじめに本実施例においてスレッド生成を行なう場合の基本的な動作について説明する。プロセッサ101の命令パイプライン105が、スレッド生成のフォーク命令を実行すると、フォークメッセージ生成装置104によってフォークメッセージが生成され、メッセージ伝達線106を介して、フォークメッセージ内で指定されたプロセッサ101のフォークメッセージキュー

ー記憶装置103に送られる。プロセッサ101は現在実行しているスレッドの実行が終了もしくは中断した時に、フォークメッセージ記憶装置103からフォークメッセージを取り出し、それに従って新たなスレッドの実行を開始する。

【0027】次に、本実施例におけるスレッド管理手段121について説明する。スレッド管理手段121はプロセッサ毎の負荷情報カウンタ108から構成される負荷状態テーブル109と解読装置110から構成される。スレッド管理手段121は各プロセッサのスレッドの実行情報を実行情報伝達線112によって集めることによって、該当するプロセッサの負荷状態カウンタ108の値を更新する。全プロセッサの負荷情報は、負荷状態テーブル109を各プロセッサ101が負荷情報伝達線111を介して読み出すことによって調べることが可能である。プロセッサ101はこの情報を用いて、負荷の軽い他のプロセッサ101に対してスレッド生成を要求するフォーク命令を実行することにより、負荷分散を行なうことが可能である。

【0028】なお、本実施例ではプロセッサ2個の並列システムについて図示したが、プロセッサ数は2個に限定するものではなく、これ以上のプロセッサがシステム中に存在する場合も同様である。

【0029】図2は、図1に示したプロセッサシステムの第1の実施例におけるスレッド管理手段についてその第2の実施例を示したブロック構成図である。本実施例は請求項2～4に対応するものである。図1においてスレッド管理手段221は第1の実施例におけるスレッド管理手段121に、負荷状態テーブル109の更新をロックするためのロック装置201を追加したものである。このロック装置201は、実行情報伝達線112を介した各プロセッサ101からの要求によって、負荷状態テーブル109の更新禁止と更新許可の設定を行なうことができる。更新禁止状態にある時は、更新禁止を要求した以外のプロセッサ101から負荷状態テーブル109を更新することはできなくなる。

【0030】請求項2では、プロセッサ101の命令にカウンタロック命令を加え、この命令を実行した場合には実行情報伝達線112にロック要求を出す。解読装置110はロック要求を解読した場合には、ロック装置201が既にロック状態にない場合には、ロック状態にセットするとともにID番号を記憶する。ただしID番号は、プロセッサ番号や命令によって指定した番号とする。既にロック状態にある場合には、このロック命令実行が失敗したという情報をロック要求を行なったプロセッサ101に返す。

【0031】請求項3では、プロセッサ101の命令にカウンタアンロック命令を加え、この命令を実行した場合には実行情報伝達線112にロック解除要求を出す。解読装置110はロック解除命令を解読した場合には、

ロック装置201が既にロック状態にあり、ロックをかけたプロセッサとロック解除要求を行なったID番号が同一である場合には、ロック状態を解除する。これ以外の場合にはロック解除命令実行が失敗したという情報をロック解除要求を行なったプロセッサ101に返す。

【0032】請求項4では、プロセッサ101でフォークを行なう場合に、フォークの必要動作と請求項3のアンロック命令を同時に行なうフォーク命令を定義する。これにより、フォークを行なった時には、ロック解除命令を実行することなく、負荷状態テーブル109のロックを解除することが可能になる。

【0033】本実施例で説明したような機能を持つカウンタロック命令、カウンタアンロック命令、フォーク命令を用いることによって以下のような動作が可能になる。まず、カウンタロック命令を実行して負荷状態カウンタ109をロックし、その値を読み出す。この値を用いてフォークによってスレッド生成を行なうプロセッサを決定してフォークを行なう。このことにより負荷分散とデータ配置を考えた並列処理が可能となる。フォークを行なう場合には、フォーク時に自動的に負荷状態カウンタ109のロックが解除されるためアンロック命令を実行する必要はない。モニタの値によってはフォークを行なわない方が得策である場合も考えられるが、この場合にはアンロック命令を実行して、フォークを行なわないで逐次処理を行なう。

【0034】図3は、図1に示したプロセッサシステムの第1の実施例におけるスレッド管理手段についてその第3の実施例を示したブロック構成図である。本実施例は請求項5によるプロセッサシステムの制御方法に対応するものである。本実施例において、スレッド管理手段321は図1に示したスレッド管理手段121に、加算器301、減算器302、選択装置303を追加したものである。

【0035】本実施例のスレッド管理手段321は、スレッドの実行開始実行を実行情報伝達線112によって受けとる。この情報は解読装置110によって解読され、そのプロセッサ101に対応する負荷情報カウンタ108の値を選ぶように選択装置303に指令を与える。加算器301によって、負荷状態カウンタ108の値に実行が開始されたスレッド数を加え、その値を負荷状態カウンタ108の新しい値とする。同様に、スレッド実行終了情報が伝えられた場合には、減算器302を用いて該当するプロセッサの負荷情報カウンタ108を減ずる処理を行なう。

【0036】負荷情報テーブル109の値は、負荷情報伝達線111を介して各々のプロセッサが参照することが可能である。これらの手段によって、システム中の全てのプロセッサの負荷情報、スレッド実行状態を各プロセッサが得ることが可能となる。

【0037】図4は、図1に示したプロセッサシステム

の第1の実施例におけるスレッド管理手段についてその第4の実施例を示したブロック構成図である。また、図5は図4に対応して、第1の実施例におけるプロセッサに関してその第2の実施例を示したブロック構成図である。これらの実施例は本発明の請求項6によるプロセッサの制御方法に対応するものである。

【0038】図4において本実施例のスレッド管理手段421は図3に示した実施例におけるスレッド管理手段302に、フォーク可否決定装置401、負荷状態基準装置402を付加し、更に図2に示した実施例におけるスレッド管理手段221内のロック装置201を付加したものである。また、図5において、本実施例のプロセッサ501は図1に示した実施例におけるプロセッサ101にフォーク可否問い合わせ装置502を追加し、更に命令パイプライン102に逐次化機能を追加して命令パイプライン503にしたものである。

【0039】次に、これらの対応する実施例の動作を順を追って説明する。命令パイプライン503においてフォーク命令が実行されると、フォーク可否問い合わせ装置502はスレッド管理手段121に対してフォーク可否の問い合わせを情報伝達線403を介して行なう。スレッド管理手段121は、フォーク可否決定装置401に対してプロセッサ501が指定したフォーク先のプロセッサへのフォークの可否を問い合わせるとともに、負荷状態テーブル109の更新を禁止する。フォーク可否決定装置401は、負荷状態テーブル109の値とフォーク可能な基準が示される負荷状態基準装置402の値を比較することによって、フォークの可否を調べ、情報伝達線404にその結果を流す。プロセッサ501には情報伝達線404によってフォークの可否判断結果が伝えられる。フォーク可能な場合にはフォークメッセージ生成装置104によってフォークメッセージが生成される。フォーク不可能な場合には、その情報が情報伝達線504によって命令パイプライン503に伝えられる。この場合、命令上はフォーク命令を実行したことになるが、命令パイプライン503上ではサブルーチンコールと同等に扱われ、自動的に逐次処理が行なわれる。これらのことにより、フォークが不可能な場合でも、フォーク命令によって自動的に逐次化可能となるため、命令パイプライン503で実行されるプログラムコードは、フォーク前にフォークの可否判断やフォーク失敗時の例外処理について考慮する必要がなくなり、実行効率向上とコードサイズ減少が可能である。このように、本実施例におけるフォーク命令は負荷状態に応じて自動的に逐次化されるという新しい制御方法を備えたフォーク命令である。

【0040】図6は、本発明によるプロセッサシステムの第2の実施例を示すブロック構成図である。本実施例は請求項7のプロセッサの制御方法に対応する。

【0041】図6において、本実施例のプロセッサシ

13

テムは図1に示したプロセッサシステムの第1の実施例に準じているが、第1の実施例におけるスレッド管理手段121に、図2のロック機構201、図3の加算器301、減算器302、選択装置303を加え、更にフォーク命令によって新たに生成されるスレッドの生成先プロセッサ番号を決定する機構を加えてスレッド管理手段621としている。なお、生成先プロセッサ番号を決定する機構は、フォーク先決定装置601とフォーク命令調停装置602から構成される。なお、本実施例におけるフォーク命令は、自動逐次化を行なうフォーク命令でも行なわないフォーク命令でもどちらでも良い。

【0042】次に、本実施例の動作を順を追って説明する。プロセッサ101の命令パイプライン105においてフォーク命令が実行されると、フォーク先を指定しないフォークメッセージをフォークメッセージ生成装置104で生成し、スレッド管理機構621に情報伝達線605を用いて送る。フォークメッセージは命令調停装置602で調停が行なわれ、フォーク先決定装置601に対してフォーク先決定要求を行なう。これとともに、フォーク先決定を行なっている間、負荷状態テーブル109の更新を禁止する。フォーク先決定装置601は、各プロセッサの負荷状態カウンタ108の値を比較することによってフォーク先のプロセッサ番号を決定する。この結果を指定されたプロセッサ101のフォークメッセージ記憶装置103に情報伝達線606を介して送るとともに、該当するプロセッサの負荷状態カウンタ108の値を加算する。これらの機構により、プロセッサはスレッド生成先の指定を行なうことなく、負荷分散を考慮したスレッドフォーク命令を実行することが可能になる。

【0043】図7は、本発明におけるプロセッサシステムの第3の実施例を示すブロック構成図である。本実施例は請求項8のプロセッサの制御方法に対応する。図7において、本実施例のプロセッサシステムは図1に示したプロセッサシステムの第1の実施例に準じているが、第1の実施例におけるスレッド管理手段121に、図2のロック機構201、図3の加算器301、減算器302、選択装置303を加え、更にスレッド確保命令によってスレッドをあらかじめ確保する機構を加えてスレッド管理手段721としている。なお、スレッドをあらかじめ確保する機構は、スレッド確保判断装置701とスレッド確保命令調停装置702から構成される。

【0044】次に、本実施例の動作を順を追って説明する。プロセッサ731の命令パイプライン105においてスレッド確保命令が実行されると、スレッド管理機構721にスレッド確保要求を情報線703を用いて送る。スレッド確保要求メッセージは命令調停装置702で調停が行なわれ、スレッド確保判断装置701に対してスレッドが確保できるか問い合わせる。これとともに、問い合わせを行っている間、負荷情報テーブル10

14

9の更新を禁止する。スレッド確保判断装置701は、各プロセッサの負荷状態カウンタ108の値を比較することによってスレッド確保が可能か判断し、結果を情報源704に流すとともに、確保できる場合には該当するプロセッサの負荷情報カウンタ108の値を更新する。これにより、フォーク命令を実行する前にあらかじめスレッドを確保できるので、最適なフォークが可能となる。

【0045】

【発明の効果】本発明によるプロセッサシステムにおいては、図1の実施例に示したように、複数のプロセッサ間でスレッド管理手段を共有している。このスレッド管理手段は各プロセッサの負荷状態を監視する負荷状態カウンタを有しており、プロセッサがこの負荷状態カウンタをアクセスすることにより、以下のように効率的に動的な並列性のコントロールを行なうことができる。

【0046】・カウンタロック命令とカウンタアンロック命令により、各プロセッサによる負荷状態カウンタの排他的なアクセスが可能になる。すなわち、カウンタロック命令によるあるプロセッサが負荷状態カウンタをロックした後これを参照し、負荷状態カウンタの値によって動的にフォークを行なうかどうかを決定するという動作が可能になる。フォークを行なわなかった場合は、カウンタアンロック命令により負荷状態カウンタのロックを解除する。このようにしてソフトウェア制御によって負荷状態に応じた並列性抽出を行なうことが可能となる。

【0047】・あるプロセッサに対応する負荷状態カウンタの値を、このプロセッサにスレッドがフォークされた際にこれを増加させ、このプロセッサがスレッドの実行を終了する際に減ずるようにすることで、簡単に負荷状態カウンタが動的な負荷状態を表すようにすることができる。

【0048】・より高度なフォーク命令の機能として、フォーク命令が発行された際にスレッド管理手段を参照し、フォークが可能かどうかを負荷状態カウンタの値から判断し、フォークが不可能な場合は当該スレッドを自動的に逐次実行することにより、マルチスレッドによる並列処理をより高性能化することが可能となる。このようなフォーク命令による動作は、ソフトウェア制御ではなく、ハードウェア制御により負荷状態に応じた動的な並列性制御を行なっていることにあたる。

【0049】・従来技術のフォーク命令は、スレッドをフォークする相手先のプロセッサを指定していた。本発明のスレッド管理手段を用いることにより、負荷状態の軽いプロセッサにスレッドをフォークするようにフォーク命令の制御方法を変更し、より柔軟な並列処理が可能となる。

【0050】・あるスレッドをフォークする前に、そのスレッドのフォークが可能かどうかあらかじめ知ってお

15

いた方がプログラム実行上の効率が良くなる場合がある。スレッド確保命令によりスレッド管理手段をアクセスし、スレッドの実行が可能かどうかを確認し、可能な場合はあらかじめ負荷状態カウンタの値を増加させ、その後実際のフォーク命令を実行することにより、このような場合についても効率的なプログラミングが可能となる。なお、スレッド確保命令の後に用いられるフォーク命令については負荷状態カウンタの値を増加させる機能は必要ない。

【0051】以上のように本発明によるプロセッサ及びプロセッサシステムにより、簡便なハードウェアで動的な並列性のコントロールを行ない、スレッドを用いた並列処理を効率化することができる。

【図面の簡単な説明】

【図1】本発明によるプロセッサ及びスレッド管理手段を含むプロセッサシステムの第1の実施例を示すブロック構成図である。

【図2】スレッド管理手段の第2の実施例を示すブロック構成図である。

【図3】スレッド管理手段の第3の実施例を示すブロック構成図である。

【図4】スレッド管理手段の第4の実施例を示すブロック構成図である。

【図5】プロセッサの第2の実施例を示すブロック構成図である。

【図6】プロセッサシステムの第2の実施例を示すブロック構成図である。

【図7】プロセッサシステムの第3の実施例を示すブロック構成図である。

【図8】従来技術のマルチスレッドアーキテクチャに基づくプロセッサ及びプロセッサシステムの例を示すブロック構成図である。

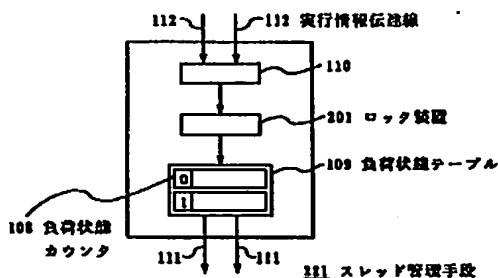
【符号の説明】

101 プロセッサ
103 フォークメッセージ記憶装置
104 フォークメッセージ生成装置
105 命令パイプライン
106 メッセージ伝達線
107 メッセージ伝達線

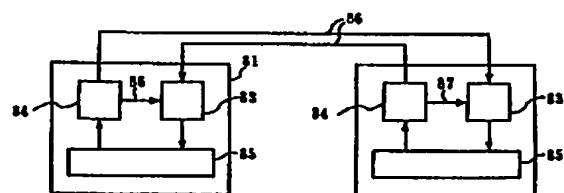
16

108 負荷状態カウンタ
109 負荷状態テーブル
110 メッセージ解読装置
111 負荷情報伝達線
112 実行情報伝達線
121 スレッド管理手段
201 ロック装置
221 スレッド管理手段
301 加算器
302 減算器
303 選択装置
321 スレッド管理手段
401 フォーク可否決定装置
402 負荷状態基準装置
403 情報伝達線
404 情報伝達線
421 スレッド管理手段
501 プロセッサ
502 フォーク可否問い合わせ装置
503 逐次化機能付命令パイプライン
504 情報伝達線
601 フォーク先決定装置
602 フォーク命令調停装置
605 情報伝達線
606 情報伝達線
621 スレッド管理手段
631 プロセッサ
701 スレッド確保判断装置
702 スレッド確保命令調停装置
703 情報伝達線
704 情報伝達線
721 スレッド管理手段
731 プロセッサ
81 プロセッサ
83 フォークメッセージ記憶装置
84 フォークメッセージ生成装置
84 命令パイプライン
86 メッセージ伝達線

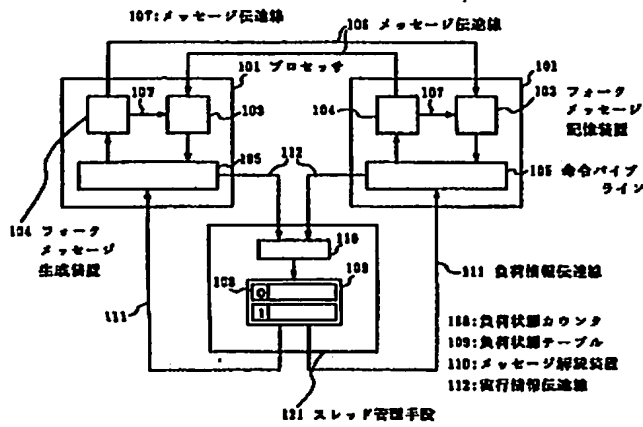
【図2】



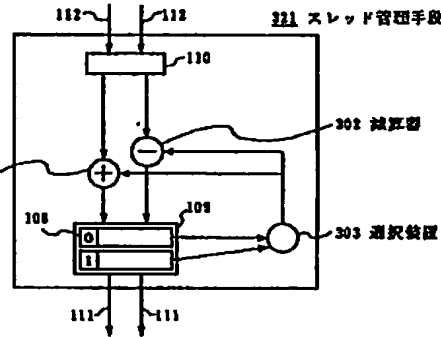
【図8】



【 図1 】

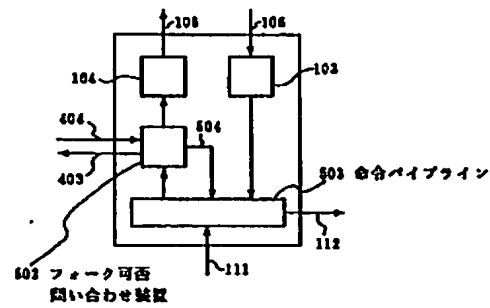


【 図3 】

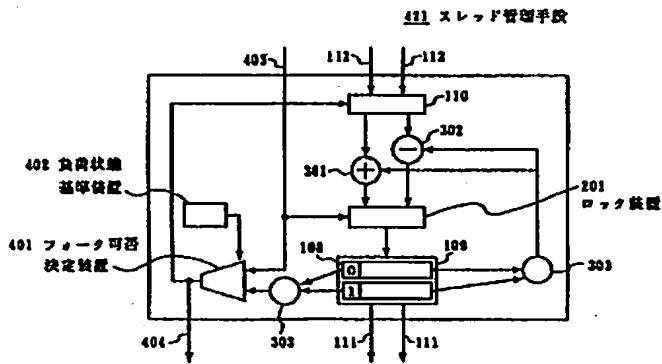


【 図5 】

501 プロセッサ



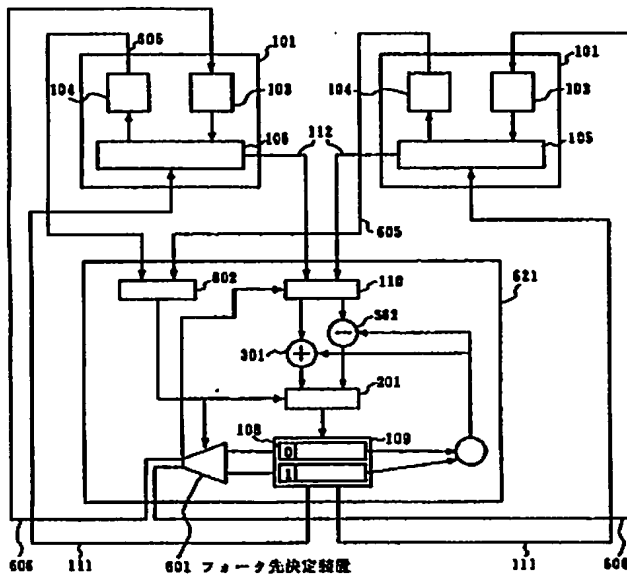
【 図4 】



【 図6 】

【 図7 】

602: 命令調停装置



702: スレッド確保命令調停装置

